In the Claims:

Please amend claims 1, 2, 4, 6, 15, 16, 18, 19, 24-30, 35 and 36 as indicated below.

1. (Currently amended) A system, comprising:

a processor; and

memory coupled to the processor and configured to store program instructions executable by the processor to implement a software documentation generator configured to:

input a plurality of sources related to a software program, wherein the plurality of sources comprises a plurality of different types of sources and comprises one or more of a software library documentation file and software program source code;

analyze the one or more sources to identify a type of each of the sources;

extract information from the plurality of sources based on the type of the source;

aggregate the extracted information into a uniform format; and

transform the aggregated information into one or more specified sets of software documentation for the software program.

2. (Currently amended) The system as recited in claim 1, wherein one of the sources comprises [[a]] the software library documentation file.

3. (Original) The system as recited in claim 2, wherein the software library documentation file comprises a tag library descriptor (TLD).

4. (Currently amended) The system as recited in claim 1, wherein one of the sources comprises ~~application~~ the software program source code.

5. (Original) The system as recited in claim 1, wherein the documentation sets comprise documentation for one or more application programming interfaces (API) provided by a software library.

6. (Currently amended) The system as recited in claim 1, wherein the software documentation generator is configured to include one or more input source plug-ins, wherein each input source plug-in corresponds to one or more of the source file types and each input source plug-in is configured to extract information from source files of types to which the plug-in corresponds.

7. (Original) The system as recited in claim 6, wherein an input source plug-in is configured to generate information not included in the corresponding source file.

8. (Original) The system as recited in claim 6, wherein each input source plug-in is configured to output data in the uniform aggregate format.

9. (Original) The system as recited in claim 6, wherein the software documentation generator is configured to include one or more transformer plug-in sets, wherein each transformer plug-in set corresponds to one or more types of output software documentation sets and each transformer plug-in set is configured to generate one or more output software documentation sets of types to which the plug-in corresponds.

10. (Original) The system as recited in claim 9, wherein the input source plug-ins are configured to produce a uniformly formatted aggregate input document and wherein

each transformer plug-in set is configured to input data included in the uniformly formatted aggregate input document.

11. (Original) The system as recited in claim 10, wherein a transformer plug-in set is configured to generate information not included in the uniformly formatted aggregate input document.

12. (Original) The system as recited in claim 1, wherein the output software documentation sets comprise one or more text files.

13. (Original) The system as recited in claim 1, wherein the output software documentation sets comprise one or more portable document files (PDF).

14. (Original) The system as recited in claim 1, wherein the output software documentation sets comprise one or more hypertext markup language (HTML) files.

15. (Currently amended) A method, comprising

receiving information from multiple sources related to a software program, wherein the multiple sources comprise a plurality of different types of sources and comprise one or more of a software library documentation file and software program source code;

extracting documentation data from said sources;

aggregating the extracted documentation data in a uniform format; and

transforming the uniformly formatted documentation data into one or more specified documentation sets for the software program;

16. (Currently amended) The method as recited in claim 15, wherein one of the sources comprises [[a]] the software library documentation file.

17. (Original) The method as recited in claim 16, wherein the software library documentation file comprises a tag library descriptor (TLD).

18. (Currently amended) The method as recited in claim 15, wherein one of the sources comprises ~~application~~ the software program source code.

19. (Currently amended) The method as recited in claim 15, wherein the documentation sets comprise documentation for one or more application programming interfaces (APIs) provided by a software library.

20. (Original) The method as recited in claim 15, wherein said extracting comprises:

analyzing a source for type and data format; and

selecting a corresponding input source plug-in based on said analyzing.

21. (Original) The method as recited in claim 15, wherein said aggregating comprises:

if a uniformly formatted aggregate input document specifies information not comprised in data extracted from the source, generating said information; and

generating a uniformly formatted aggregate input document.

22. (Original) The method as recited in claim 15, wherein said transforming comprises:

selecting a transformer plug-in set corresponding to a specified output documentation format; and

translating a portion of a uniformly formatted aggregate input document into one or more elements of a software documentation set in the specified output documentation format.

23. (Original) The method as recited in claim 15, wherein the output software documentation sets comprise one or more text files.

24. (Currently amended) The method as recited in claim 15, wherein the output software documentation sets comprise one or more portable document files (PDFs).

25. (Currently amended) The method as recited in claim 15, wherein the output software documentation sets comprise one or more hypertext markup language (HTML) files.

26. (Currently amended) A computer-accessible memory medium storing ~~comprising~~ program instructions, wherein the program instructions are computer-executable to:

input a plurality of sources related to a software program, wherein the plurality of sources comprises a plurality of different types of sources and comprises one or more of a software library documentation file and software program source code;

analyze the one or more source files to identify the type of each of the source files;

extract information from the plurality of sources based on the type of the source;

aggregate the extracted information into a uniform format; and

transform the aggregated information into one or more specified sets of software documentation for the software program.

27. (Currently amended) The computer-accessible medium as recited in claim 26, wherein one of the sources comprises [[a]] the software library documentation file.

28. (Currently amended) The computer-accessible medium as recited in claim 27, wherein the software library documentation file comprises a tag library descriptor (TLD).

29. (Currently amended) The computer-accessible medium as recited in claim 26, wherein one of the sources comprises ~~application~~ the software program source code.

30. (Currently amended) The computer-accessible medium as recited in claim 26, wherein the documentation sets comprise documentation for one or more application programming interfaces (APIs) provided by a software library.

31. (Original) The computer-accessible medium as recited in claim 26, wherein to extract comprises to:

analyze a source for type and data format; and

select a corresponding input source plug-in based on said analyzing.

32. (Original) The computer-accessible medium as recited in claim 26, wherein to aggregate comprises to:

if a uniformly formatted aggregate input document specifies information not comprised in data extracted from the source, generate said information; and

generate the uniformly formatted aggregate input document.

33. (Original) The computer-accessible medium as recited in claim 26, wherein to transform comprises to:

select a transformer plug-in set corresponding to a specified output documentation format; and

translate a portion of a uniformly formatted aggregate input document into one or more elements of a software documentation set in the specified output documentation format.

34. (Original) The computer-accessible medium as recited in claim 26, wherein the output software documentation sets comprise one or more text files.

35. (Currently amended) The computer-accessible medium as recited in claim 26, wherein the output software documentation sets comprise one or more portable document files (PDFs).

36. (Currently amended) The computer-accessible medium as recited in claim 26, wherein the output software documentation sets comprise one or more hypertext markup language (HTML) files.